Tracking of Dynamical Processes with Model Switching Using Temporal Convolutional Networks

Arick Grootveld, Vlad I. Bugayev, Leah Lackey, Andrew G. Klein
Department of Engineering and Design
Western Washington University
Bellingham, WA 98225
{grootva, bugayev, lackeyl, andy.klein}@wwu.edu

Kirty Vedula, D. Richard Brown III
Department of ECE
Worcester Polytechnic Institute
Worcester, MA 01609
{kpvedula, drb}@wpi.edu

***Abstract*—This paper considers the problem of tracking and predicting dynamical processes with model switching. The classical approach to this problem has been to use an interacting multiple model (IMM) which uses multiple Kalman filters and an auxiliary system to estimate the posterior probability of each model given the observations. More recently, data-driven approaches such as recurrent neural networks (RNNs) have been used for tracking and prediction in a variety of settings. An advantage of data-driven approaches like the RNN is that they can be trained to provide good performance even when the underlying dynamic models are unknown. This paper studies the use of temporal convolutional networks (TCNs) in this setting since TCNs are also data-driven but have certain structural advantages over RNNs. Numerical simulations demonstrate that a TCN matches or exceeds the performance of an IMM and other classical tracking methods in two specific settings with model switching: (i) a Gilbert-Elliott burst noise communication channel that switches between two different modes, each modeled as a linear system, and (ii) a maneuvering target tracking scenario where the target switches between a linear constant velocity mode and a nonlinear coordinated turn mode. In particular, the results show that the TCN tends to identify a mode switch as fast or faster than an IMM and that, in some cases, the TCN can perform almost as well as an omniscient Kalman filter with perfect knowledge of the current mode of the dynamical system.**

TABLE OF CONTENTS

1. INTRODUCTION 1
2. SYSTEM MODEL 2
3. STATE ESTIMATION AND PREDICTION METHODS .. 3
4. TEMPORAL CONVOLUTIONAL NETWORK 4
5. RESULTS 5
6. CONCLUSION 7
ACKNOWLEDGMENTS 7
REFERENCES 7
BIOGRAPHY 8

1. INTRODUCTION

In many practical dynamical systems, parameters vary, operation modes change, and failures occur, leading to changes in the system dynamics [1]. Often, such stochastic systems are modeled by switching between multiple dynamic models. The optimal state estimator for such model switching problems is either unknown or prohibitively complex, however, and often requires perfect knowledge of all the underlying multiple models. Perhaps the most tractable, studied class of such systems is the case where the multiple models are linear and the switching between models is Markovian, resulting in so-called Markov jump linear systems (MJLSs). Even within this class of systems with model switching, however, the optimal estimator is computationally impractical in general, and the widely adopted approach for state estimation is to resort to suboptimal multiple-model filtering algorithms such as the interacting multiple model (IMM) [2] which explicitly assumes the multiple models are known.

This paper investigates a data-driven approach to estimation and prediction of systems with model switching which differs from the majority of classical estimation approaches in that it does not require explicit knowledge of the multiple models and makes use of very recent advances in machine learning for so-called "sequence prediction" problems. While recurrent neural networks (RNNs) and long short-term memory (LSTM) have historically been the most popular choice for sequence prediction, recent results in machine learning have shown that a new architecture called temporal convolutional networks (TCNs) can often outperform LSTM-based RNNs in sequence prediction problems [3]. In addition to their performance advantage in a wide range of sequence prediction problems, TCNs have a number of implementation advantages over RNNs in that they admit a more parallelizable implementation and have reduced memory requirement for training; in addition, TCNs have several structural advantages such as an adjustable receptive field size, and avoidance of the problem of exploding/vanishing gradients known to plague RNNs.

The performance of the proposed TCN approach to state prediction for dynamical systems with model switching is demonstrated in two practical applications:

1. *Communication over Gilbert-Elliott Channels*: In practical communication systems, burst errors can arise on the channel thus making communication challenging. A simple but effective model for simulating channel-induced burst errors is the Gilbert-Elliott two state Markov model [4, 5], which uses a Markov chain with two states representing a good channel mode and a bad channel mode. In this model the channel transitions between these two modes, depending on the current state of the model. We consider a time-varying narrowband channel that switches periodically between two distinct autoregressive channel models.

2. *Maneuvering Target Tracking*: Maneuvering target tracking estimates or predicts aircraft motion, often by a radar or other detection and ranging sensor to control its flight path, or direct its movement in accordance with other operations. Aircraft motion is typically modeled as switching between two or more linear systems, with a different linear system representing constant velocity, constant acceleration, coordinated turns, or other possible maneuvers. For a comprehensive survey of maneuvering target models, we point the reader to [6]. We consider a scenario where a target switches between a constant velocity mode and a coordinated turn mode.

2021 IEEE Aerospace Conference | 978-1-7281-7436-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/AERO50100.2021.9438450

Numerical results are presented to demonstrate the performance of the proposed TCN-based approach in both of these examples of dynamical systems with model switching. The optimal estimator for linear dynamics in the absence of switching, e.g., a single autoregressive channel or a target with constant velocity, is the Kalman filter. Thus, as an initial validation step, simulations are used to demonstrate the TCN-based approach performs as well as the optimal Kalman filter for both applications in the absence of switching. The switching scenario is then considered, and simulations demonstrate the effectiveness of the TCN-based approach compared with an omniscient Kalman filter which has perfect knowledge of the current operation mode and the underlying linear system model and noise covariances of each mode. The performance of the proposed TCN-based approach is also compared with other popular suboptimal estimators such as the Least Squares (LS) and IMM filters. The results from both applications demonstrate that the TCN-based approach achieves performance close to optimal without knowledge of the system dynamics or noise statistics. Moreover, the proposed approach is sufficiently general in that it can be applied to a wide variety of dynamical systems with model switching.

2. SYSTEM MODEL

We consider a framework for dynamical systems that undergo switching in time among several modes or sub-systems. Such dynamical systems have a long history in control theory, and are sometimes referred to as "jump systems" [7]. Under a fairly general discrete-time framework, the system state and measurement evolve according to

$$\begin{aligned} x[k+1] &= f_{\theta_k}(k, x[k], v[k]) \\ y[k] &= h_{\theta_k}(k, x[k], w[k]) \end{aligned}$$

where $x[k]$ is the state vector, $y[k]$ is the measurement vector, $v[k]$ is the random process noise, $w[k]$ is the random measurement noise, $f_{\theta_k}(\cdot)$ is a family of N vector functions describing the state dynamics during mode θ_k, $h_{\theta_k}(\cdot)$ is a family of N vector functions describing the measurement dynamics, and $\theta_k \in \{0, 1, \ldots, N-1\}$ denotes the stochastic system mode in effect during the sample period ending at discrete time k. We assume throughout that the current system mode θ_k in effect is not known, though the statistics of the random process may be known.

Under certain assumptions about the model dynamics and the stochastic switching, several important special cases of this system model emerge. When the switching process θ_k can be described by a Markov chain with time-invariant transition matrix P, so-called Markov Jump Systems arise [8]. If, in addition, the system is linear, the system is called a Markov Jump Linear System (MJLS) [1] and it admits the state space realization

$$x[k+1] = F_{\theta_k}[k]x[k] + v[k] \quad (1)$$

$$y[k] = H_{\theta_k}[k]x[k] + w[k] \quad (2)$$

where the non-linear functions $f_{\theta_k}(\cdot)$ and $h_{\theta_k}(\cdot)$ are replaced by the matrices $F_{\theta_k}[k]$ and $H_{\theta_k}[k]$. When the process and measurement noises can be modeled as Gaussian random processes, we assume they are distributed as $v[k] \sim \mathcal{N}(0, Q)$ and $w[k] \sim \mathcal{N}(0, R)$, respectively, with Q and R as the corresponding covariance matrices. Again, while we assume that the mode θ_k in effect at time k is not known, at times we will assume knowledge of the transition matrix P that completely characterizes the statistics of the underlying Markov chain.

Having described the general model and notation that we will use throughout this paper, we now discuss specific system models for two applications within this class of switching systems: (i) communication through a time-varying Gilbert-Elliott channel which models switching between two different modes, and (ii) tracking maneuvering targets that switch between a near constant velocity mode and a coordinated turn mode.

Gilbert-Elliott Channel

A Gilbert-Elliott channel models communication through narrowband channels that tend to induce burst errors in the received data. Generally, these bursts of contiguous erroneous symbols arise when the channel makes a jump from a "good" mode to a "bad" mode. Thus, such a channel model has $N = 2$ modes where $\theta_k = 0$ represents the situation where the system is in the "good" channel mode, and $\theta_k = 1$ represents the "bad" channel mode that induces burst errors.

In this paper, the unknown baseband channel gain is assumed to be complex, representing in-phase and quadrature components. In addition, we assume that the channel is time-varying according to an autoregressive (AR) process, and the measurement $y[k]$ represents noisy observations of the complex channel gain. While the channel gain itself is time-varying, the family of matrices F_{θ_k} governing the AR system dynamics for each mode is assumed to be static in this application. Specifically, the scalar complex channel gain $c[k]$ at time k is modeled by an mth-order AR process

$$c[k] = \sum_{j=1}^{m} a_{j,\theta_k} c[k-j] + v'[k] \quad (3)$$

where a_{j,θ_k} denotes the AR parameters of mode θ_k for $j \in \{1, \ldots, m\}$ and $v'[k]$ is an i.i.d. zero-mean complex Gaussian scalar process with variance σ_v^2. Because mode 0 represents the "good" channel mode, the AR coefficients $a_{j,0}$ corresponding to this mode are expected to lead to more reliable communication than the set of AR coefficients $a_{j,1}$ corresponding to the "bad" mode 1. For example, the mode 0 coefficients might be easier to estimate or might lead to higher average received signal-to-noise ratio.

Putting the AR model in the dynamical systems framework above, this leads to a MJLS governed by these equations:

$$\begin{aligned} x[k+1] &= \left[\begin{array}{ccc|c} a_{1,\theta_k} & a_{2,\theta_k} & \cdots & a_{m,\theta_k} \\ & & & 0 \\ & I_{m-1} & & \vdots \\ & & & 0 \end{array}\right] x[k] + v[k] \\ y[k] &= \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} x[k] + w[k] \end{aligned}$$

where the state vector is defined in terms of the scalar AR process above as $x[k] = [c[k-1] \cdots c[k-m]]^\top \in \mathbb{C}^m$ and I_{m-1} is the identity matrix of size $m-1$. The process noise is distributed as $v[k] \sim \mathcal{CN}(0, Q)$, though in this case Q is all zeros except for the top left corner which equals $[Q]_{0,0} = \sigma_v^2$, so that process noise is only added to the topmost state, making the model match the scalar AR process model (3). The observation $y[k]$ is a complex scalar, and the measurement noise is distributed as $w[k] \sim \mathcal{CN}(0, \sigma_w^2)$. Finally, the statistics of the Markov mode switching are

uniquely defined by the initial state probabilities and the state transition matrix P, as depicted in Fig. 1, where the elements of the matrix P are given by p_{ij}.

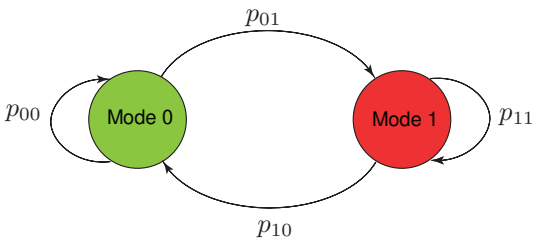


Figure 1. Two-state Markov chain modeling switching in the Gilbert-Elliott channel

Maneuvering Target Tracking

In maneuvering target tracking, the primary objective is using noisy measurements of a moving object (acquired, for example, via radar) to estimate or predict state trajectories. The body of literature concerned with dynamical models of target tracking is rich, and there are numerous models for describing the dynamics of target motion (see [6] for a comprehensive survey). Target motions generally fall into two classes: non-maneuvering and maneuvering. A non-maneuvering motion is uniform motion at a nearly constant velocity, whereas a maneuvering motion is most any other motion. Here, we consider a popular discrete-time maneuvering target tracking model [2] that switches between $N = 2$ such modes. Specifically, we assume that the target is either in a near constant velocity (CV) mode or in a coordinated turn (CT) mode. Moreover, we again assume that the switching dynamics are described by a two-state Markov chain so that the mode switching model shown in Fig. 1 applies to this application, as well. While the model for CV mode turns out to be linear, the CT mode obeys a nonlinear model. We now describe the dynamical model for each of the two modes for state dimension compatibility when switching between modes.

Constant Velocity Model—The model for CV mode has four states $x[k] = [\xi[k]\ \dot{\xi}[k]\ \eta[k]\ \dot{\eta}[k]]^\top$ with ξ and η denoting Cartesian coordinates in the horizontal plane, and $\dot{\xi}$ and $\dot{\eta}$ denoting the velocity. The model for operation in CT mode is linear and is described by

$$x[k] = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x[k-1] + \begin{bmatrix} \frac{1}{2}T^2 & 0 \\ T & 0 \\ 0 & \frac{1}{2}T^2 \\ 0 & T \end{bmatrix} v[k] \qquad (4)$$

with $v[k] \sim \mathcal{N}(0, Q)$ as in [2]. The process noise here serves to model turbulence or other non-idealities. Because the CT model below has five states, it is common to append a fifth state to this model that is always zero.

Coordinated Turn Model—The coordinated turn (CT) model has nearly constant speed and turns at a constant rate. It adds a fifth state to the CV model and is defined as $x[k] = [\xi[k]\ \dot{\xi}[k]\ \eta[k]\ \dot{\eta}[k]\ \Omega[k]]^\top$ where $\Omega[k]$ denotes the turn rate. The CT model is *nonlinear* and can be written as

$$x[k] = \begin{bmatrix} 1 & \frac{\sin(\Omega[k]T)}{\Omega[k]} & 0 & -\frac{1-\cos(\Omega[k]T)}{\Omega[k]} & 0 \\ 0 & \cos(\Omega[k]T) & 0 & -\sin(\Omega[k]T) & 0 \\ 0 & \frac{1-\cos(\Omega[k]T)}{\Omega[k]} & 1 & \frac{\sin(\Omega[k]T)}{\Omega[k]} & 0 \\ 0 & \sin(\Omega[k]T) & 0 & \cos(\Omega[k]T) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} x[k-1] \qquad (5)$$

$$+ \begin{bmatrix} \frac{1}{2}T^2 & 0 & 0 \\ T & 0 & 0 \\ 0 & \frac{1}{2}T^2 & 0 \\ 0 & T & 0 \\ 0 & 0 & 1 \end{bmatrix} v[k].$$

Because the matrix multiplying $x[k-1]$ in the above equation is a nonlinear function of one of the state variables, $\Omega[k]$, it is clear that this is a nonlinear model. In models where the turn rate $\Omega[k]$ is known and therefore not a state variable, this CT model reduces to a linear system.

The model for the measurement $y[k]$ in both CT mode as well as the CV mode (with an appended fifth zero state, as mentioned above) is given by

$$y[k] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} x[k] + w[k]$$

where $w[k] \sim \mathcal{N}(0, R)$. That is, the measurement consists of noisy observations of $\xi[k]$ and $\eta[k]$, i.e., the Cartesian coordinates of the target.

3. State Estimation and Prediction Methods

We now briefly review several classical methods for estimating and predicting states in dynamical systems. While our primary concern in this paper is prediction, estimation and prediction methods are very tightly intertwined. Under certain assumptions, several of these methods are optimal estimators in non-switching dynamical systems; however, all of the these estimators are suboptimal estimators when used in dynamical systems with model switching.

When the system dynamics of each mode are known, the optimal approach to estimation and prediction in models with switching generally involves building independent parallel estimators for all possible mode histories (i.e., all possible sequences θ_k). Because this is exponentially complex even for the simplest classes of dynamical models with switching (e.g., MJLSs), optimal estimators are impractical in systems with model switching. For this reason, suboptimal approaches are used exclusively in practice.

Kalman Filter

For a linear state-space model with Gaussian noise and known process and measurement noise covariances, the optimal solution is provided by the KF with the standard recursive prediction and update equations that calculate estimates and predictions [9, 10]. Since we only present the salient ideas here, the reader is directed to [11–13] for further details. For each iteration,

1. The prediction step estimates the current state estimate with the previous state estimate, $\hat{x}[k|k]$, taken forward one step via the linear transformation $\hat{x}[k+1|k] = F_{k+1}\hat{x}[k|k]$.
2. The update step performs a linear combination of $\hat{x}[k|k]$ and actual measurements $y[k]$.
3. Covariances of both state estimates are calculated during both the prediction and update stages.

A major caveat for the KF is that in order to obtain an optimal state estimate, the KF requires exact knowledge of the system model as well as the process noise and measurement noise parameters. The performance of the KF degrades if there are mismatches between the true dynamical system and the assumed model; this topic has received significant attention in the literature (e.g. [14]) and numerous approaches have been proposed to mitigate this degradation.

Least Squares Prediction

While least squares is a well-known technique, we briefly review it here because it as an example of a "data-driven" estimator that does not require knowledge of the underlying dynamical system model, though it does require training data containing a collection of known state variables and the corresponding observations. In the steady state, a Kalman filter predictor can be written in the form

$$\hat{x}[k+1|k] = a_0 y[k] + a_1 y[k-1] + a_2 y[k-2] + \dots$$

where, if the dynamics and noise parameters are all known, the $\{a_i\}$ coefficients can all be computed as functions of the steady state prediction and estimation covariances. These are all, consequently, functions of the steady state Kalman gain. This results in a stable IIR filter that resembles a Kalman filter, and it can be approximated by truncating the number of terms so that

$$\hat{x}[k+1|k] \approx a_0 y[k] + a_1 y[k-1] + ... + a_L y[k-L]$$

where $L+1$ is the number of terms in the truncated sequence. Thus the problem is to find $\{a_0, ...a_L\}$ to minimize the mean squared prediction error without requiring knowledge of the dynamics or noise parameters. Given enough $y[k]$ observations, this can be accomplished using least squares on the linear system of equations

$$\underbrace{\begin{bmatrix} \hat{x}[k] \\ \vdots \\ \hat{x}[k-M] \end{bmatrix}}_{\triangleq \hat{X}} = \underbrace{\begin{bmatrix} y[k-1] & \dots & y[k-L-1] \\ \vdots & \ddots & \vdots \\ y[k-M-1] & \dots & y[k-L-M-1] \end{bmatrix}}_{\triangleq Y} \underbrace{\begin{bmatrix} a_0 \\ \vdots \\ a_L \end{bmatrix}}_{\triangleq a}$$

which for $M \geq L$ can be solved as a standard least squares problem, i.e.,

$$a_{LS} = (Y^\top Y)^{-1} Y^\top \hat{X}.$$

Interacting Multiple Model for Maneuvering Targets

Unlike the prior estimators which do not give any special consideration to the switching nature of the model, the IMM filter [15] is a suboptimal estimator designed specifically for dynamical systems with model switching. The IMM falls into the class of estimators that use multiple filter models, typically with one matched to each of the N modes of the system. Other approaches in this class include, for example, the Generalized Pseudo-Bayesian (GPB) methods [16]. In such approaches, minimizing computation becomes very important due to the exponentially increasing number of state hypotheses. The IMM effectively combines hypotheses from multiple filter models in a computationally efficient manner, and is therefore widely used in practice for state estimation in dynamical systems with model switching. The model inaccuracy is addressed by facilitating an interaction between them for different modes at the beginning of each filter cycle. These are weighed accordingly by the conditional probabilities of switching between model modes. We summarize the algorithm briefly here and direct the reader to [2] for a detailed treatment.

The IMM estimates the blended states and covariances iteratively at each step by combining the initial conditions, states, and their associated covariances according to the mode transition probabilities. Denote the state estimate at time $k-1$ of the filter matched to the ith mode as $x^{(i)}[k-1|k-1]$ and its corresponding covariance $\Sigma^{(i)}[k-1|k-1]$ for $i \in 1, \dots, N$, then each step of the IMM filter performs the following:

1. Calculate mixing probabilities $\{\mu_{i|j}[k-1|k-1]\}_{i,j=1}^N$, mixed estimates $\{\hat{x}^{(0i)}[k-1|k-1]\}_{i=1}^N$ and covariances $\{\Sigma^{(0i)}[k-1|k-1]\}_{i=1}^N$.
2. Using each of the N mode-matched models, calculate predicted estimates $\hat{x}^{(i)}[k|k-1]$ and covariances from mixed estimates in the previous step for i^{th} model, $i \in 1, \dots, N$.
3. Calculate updated estimates $\hat{x}^{(i)}[k|k]$ and covariances from the predicted estimates for i^{th} model, $i \in 1, \dots, N$ and calculate the updated mode probabilities $\mu_i[k]$.
4. Calculate the output state estimate and covariance estimates. The overall output state estimate is computed as $\hat{x}[k|k] = \sum_{i=1}^N \hat{x}^{(i)}[k|k]\mu_i[k]$.

While the traditional IMM as outlined above outputs state estimates, it can also be used to output state predictions. By using the assumption that the predicted mode probabilities at time $k+1$ given knowledge up through time k are equal to the estimated mode probabilities at time k, i.e., that $\mu_i[k+1|k] \approx \mu_i[k]$ as in [17], the overall predicted state can be computed via

$$\begin{aligned} \hat{x}[k+1|k] &= \sum_{i=1}^N \hat{x}^{(i)}[k+1|k]\mu_i[k+1|k] \\ &\approx \sum_{i=1}^N \hat{x}^{(i)}[k+1|k]\mu_i[k]. \end{aligned}$$

4. TEMPORAL CONVOLUTIONAL NETWORK

While convolutional neural networks have shown great promise as an effective machine learning architecture, particularly in image recognition applications [18], CNNs have more recently been employed as a fundamental piece of *temporal convolutional networks* for time-series modeling and sequence prediction problems. TCNs have demonstrated state-of-the-art performance over a wide range of prediction applications, including weather prediction [19], traffic prediction [20], audio [21], and action segmentation [22] to name a few. Moreover, as mentioned in Section 1, TCNs offer several implementation advantages over LSTM-based RNNs, which have traditionally been the most widely used architecture for sequence prediction problems [3].

TCNs use dilated convolutions to increase the receptive field of the network. They are more memory efficient than recurrent networks due to the shared convolution architecture which allows long sequences to be processes in parallel. In RNNs, the input sequences are processed sequentially, which results in higher computation time. However, TCNs are trained with the standard backpropagation algorithm, hence avoiding the gradient problems of the backpropagation-through-time algorithm used in RNNs.

TCNs are generally used in one of two configurations: (i) for sequence estimation where the output sequence is the same length as the input sequence, just as with RNNs, or (ii) for autoregressive prediction of samples at some time in the future, which is typically accomplished by adding a fully connected layer to the output of the sequence prediction. In this paper, we consider the latter configuration and the corresponding architecture is shown in Fig. 2. Essentially, a TCN employs

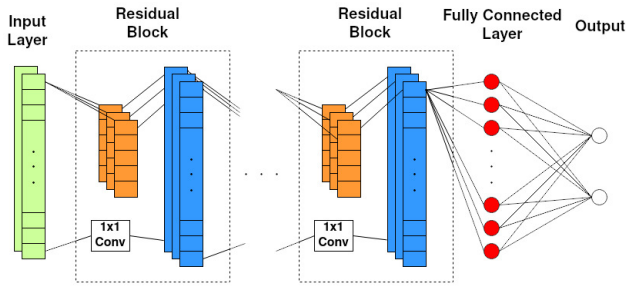


Figure 2. Schematic of a Temporal Convolutional Network with a series of residual blocks with increasing dilation followed by a fully connected layer.

a series of connected residual blocks consisting of 1D fully-convolutional networks (FCNs) [23] where the convolutions are chosen to be causal, primarily through appropriate choice of zero-padding, and dilation [22]. In TCNs, dilation takes the place of "pooling" which is commonly used in more traditional CNN architectures, and it allows the TCN to have a large receptive field.

A TCN can accept a multi-dimensional input sequence (i.e. a matrix). In the two applications considered in this paper – Gilbert-Elliott channel prediction and maneuvering target tracking – the input vectors are two-dimensional as shown in Fig. 2. The specific TCN implementation that we consider in this paper is the one described in [3], which is the basis for the Python library [24] we used. Finally, because analysis of the TCN performance as an estimator or predictor is intractable, we present simulated performance in the next section.

5. RESULTS

We now present the simulated performance of the TCN when used for prediction in two example dynamical systems with model switching. The performance of the TCN is compared to the IMM, LS, and a *Genie Kalman Filter* (GKF) which is a time-varying KF with perfect knowledge of the current mode θ_k. When the current mode θ_k is known and the mode dynamics are linear, the system reduces to a standard linear time-varying dynamical model for which the KF is optimal. Hence, for linear mode dynamics, the MSE of the GKF establishes a lower bound by which other algorithms can be compared.

Gilbert-Elliott Channel Prediction Example

We selected a 2nd-order model with AR coefficients chosen as

$$\begin{aligned}\text{Mode 0 (good)} &: a_{1,0} = 0.3,\ a_{2,0} = 0.1\\ \text{Mode 1 (bad)} &: a_{1,1} = 1.949,\ a_{2,1} = -0.95\end{aligned}$$

and the process and measurement noise were both chosen to be $\sigma_v^2 = \sigma_w^2 = 0.1$. Note here that the "bad" channel has a characteristic root very near the unit circle. The transition probabilities of the Markov chain that define the mode were set to $p_{00} = p_{11} = 0.9995$ and $p_{01} = p_{10} = 0.0005$, forming a homogeneous Markov chain. We selected these parameters so that the probability of switching would be low, such that the model would stay in either mode for an extended period of time in order to model a burst noise channel [4]. The Kalman filters matching mode 0 and mode 1 are denoted as "KF0" and "KF1", respectively. For the complete implementation of the Gilbert-Elliott channel model used here, we refer the reader to the code repository [25].

The TCN used for this problem was designed with a total of 18,102 trainable parameters comprised of two stacks of three layers consisting of 20 convolutional filters, and the kernel size was chosen to be 4. We trained a TCN with 4×10^6 samples to predict a pair of numbers representing the real and imaginary portion of the complex state of the channel. For each prediction, the TCN was provided an input containing the 10 most recent complex noisy channel observations. The complex channel observations were split into real and imaginary parts [26] such that, for each prediction, the TCN used an input from $\mathbb{R}^{10\times 2}$.

Table 1. Gilbert-Elliott channel prediction MSE.

	With mode switching	Only mode 0 ("good")	Only mode 1 ("bad")
Genie KF	**0.216**	**0.106**	**0.321**
KF0	291	**0.106**	560
KF1	0.427	0.352	**0.321**
Least Squares	0.409	0.294	0.344
IMM	0.307	**0.106**	**0.321**
TCN	0.309	**0.106**	**0.322**

Results for the Gilbert-Elliott scenario described in Table 1 were obtained from Monte Carlo simulations with 4×10^6 output pairs. The results are further broken down into performance on a model with mode switching and two models without mode switching ("only mode 0" and "only mode 1"). As expected, KF0 is optimal for a system with "only mode 0" dynamics and KF1 is optimal for a system with "only mode 1" dynamics, while both perform badly in a system with mode switching. The GKF is optimal in all cases since it has perfect knowledge of the mode. The TCN and IMM achieve nearly identical performance, and they are close to optimal in both cases without mode switching and outperforms all algorithms with switching, achieving an MSE close to that of the GKF. We again recall that the IMM requires exact knowledge of the AR channel coefficients, knowledge that is not required by the TCN. As a data-driven approach, the TCN requires the availability of significant training data.

It is notable that KF0 has poor performance when making predictions of states generated under mode 1. The large errors experienced by KF0 under mode 1 are due to the extreme coefficient mismatch between what KF0 expects and the actual model coefficients in mode 1. Intuitively, KF0 assumes that the current and previous states will have a small effect on the next prediction, which is optimal for mode 0 but far from optimal under mode 1.

As for the LS predictor, with mode switching, LS trains on a combination of mode 0 and mode 1 data. As can be seen from the performance of KF1 and KF0 under the respective mismatch scenarios, a KF with coefficients that align with the higher error rate model will perform better when model

mismatch occurs, compared to a KF with coefficients more effective under a lower error rate model. Hence, LS finds a model such that the equivalent KF would perform best across all modes. This will of course result in performance closer aligned to KF1 than KF0, since the performance of KF0 under mode 1 is so poor.

Figures 3 and 4 demonstrate the algorithms' performance from initialization to steady state averaged over 5,000 Monte Carlo simulations. Since the TCN and IMM performance is nearly identical, IMM curves have been omitted for visual clarity. Figure 3 demonstrates the mean squared prediction error performance for predictions of the first state of the AR process by the TCN, LS, and KF0 for the "only mode 0" case. These results show that the TCN closely tracks the (optimal) performance of KF0. Figure 4 similarly shows the achieved performance for the "only mode 1" case. In this case, the TCN mean squared prediction error performance lags approximately one sample behind the (optimal) performance of KF1. In both of these examples, the TCN achieves the steady state performance shown by the black dashed "Ricatti line" within a handful of samples.

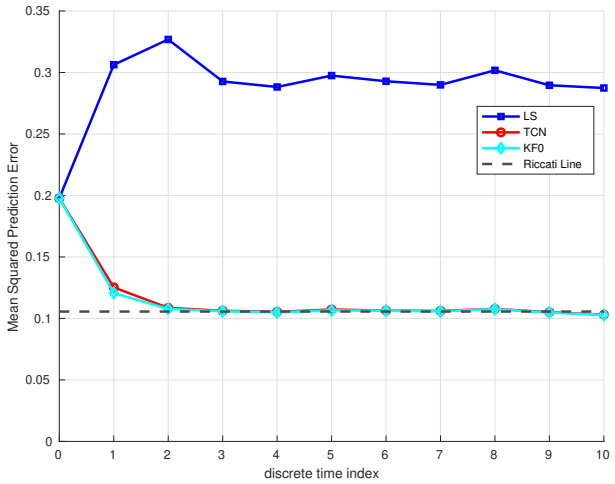


Figure 3. Gilbert-Elliott mode 0 ("good channel") prediction error, averaged over 5,000 realizations.

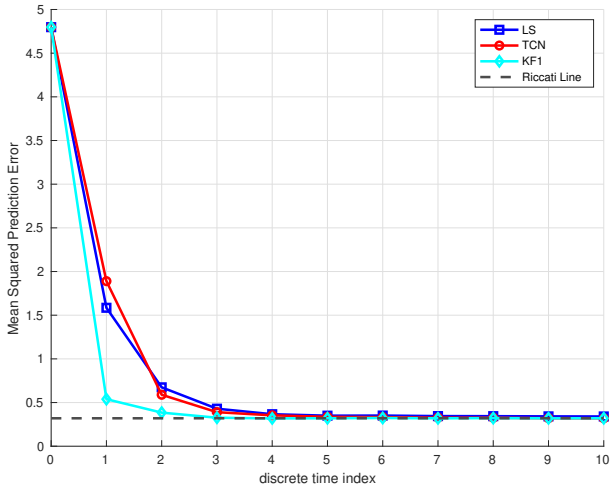


Figure 4. Gilbert-Elliott mode 1 ("bad channel") prediction error, averaged over 5,000 realizations.

Maneuvering Targets Tracking Example

Mode 0 was chosen to be the nearly constant velocity model described by (4), while mode 1 was chosen to be the (non-linear) coordinated turn model described by (5). The CV-to-CT and CT-to-CV transition probabilities were selected as $p_{01} = p_{10} = 0.01$, and the remaining system parameters were selected to be $T = 1\,\text{sec}, Q = I_2, R = 576 \cdot I_2$ where I_2 is a 2×2 identity matrix. The initial speed of the target was normally distributed with a mean of 120 m/s and a standard deviation of 30 m/s. The magnitude of the initial turn rate at the start of each turn was normally distributed with a mean of 4 rad/sec and a standard deviation of 1 rad/sec, and with left and right turns being equally likely; thus, $\Omega[k]$ at the start of each turn has a folded Gaussian distribution. We perform 4×10^4 Monte-Carlo simulations with 10^3 samples to ensure a consistent performance across multiple mode transitions. We report the average prediction RMSE in Table 2, where the RMSE is computed only over the two Cartesian coordinate states of the target, i.e., $RMSE = \sqrt{E[(\xi - \hat{\xi})^2 + (\eta - \hat{\eta})^2]}$. The performance of the algorithm is evaluated similar to the Gilbert-Elliott scenario. Any samples occurring 25 samples or fewer after a mode transition are defined as as being in a "transition regime". The CV and CT mode performance in the table indicates an average RMSE only over those time slots which are not in a transition regime, i.e., where no transition occurred within the previous 25 time slots. An example transition is shown in Figure 5.

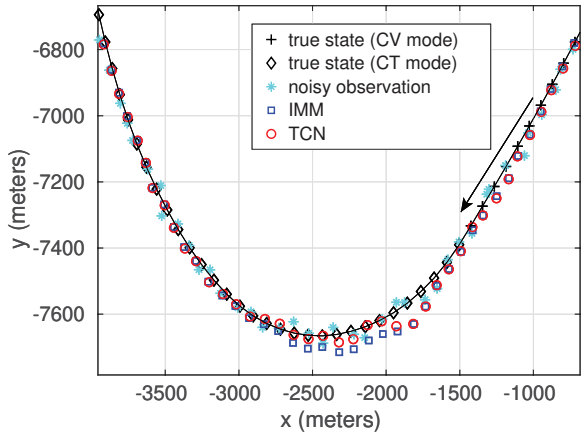


Figure 5. Example trajectory showing transition from CV to CT. Target is moving in direction of arrow, and we define the "transition" as having a duration of 25 samples.

To obtain a lower bound on the performance as in the Gilbert-Elliott scenario, we again use a Genie Kalman Filter which has knowledge of the current mode. Here, however, the GKF is additionally provided knowledge of the current turn rate $\Omega[k]$ (i.e., the fifth element in the state vector). With this knowledge, the CT model in (5) becomes a time-varying *linear* model and thus the GKF is the optimal estimator. We note that providing this additional information to the GKF may result in its prediction MSE being a rather loose lower bound for CT mode operation.

The network is trained on 4×10^6 samples with the same architecture consisting of 18,102 trainable parameters described in the prior Gilbert-Elliott section. Additional details concerning other TCN hyperparameters can be found in the code repository [25]. For each prediction, the TCN was provided the two Cartesian coordinates of the 20 most recent

noisy observations of the target, and thus the input was from $\mathbb{R}^{20\times 2}$. Just prior to the TCN input, the coordinate system of every input was translated so that each length-20 input sequence terminated at the origin; at the TCN output, this translation was reversed to put the outputs back on the original coordinate system. The reason for this translation is that data pre-processing has been shown to improve the performance of machine learning systems, particularly when the pre-processing serves to limit the inputs and outputs to a narrower or more balanced range of values.

Table 2. Maneuvering target prediction RMSE (in meters)

	With mode switching	Only mode 0 (CV)	Only mode 1 (CT)
Genie KF	**19.5**	**19.4**	**19.6**
IMM	28.0	**20.0**	**24.8**
TCN	**25.4**	20.6	25.9

The results in Table 2 show that the prediction RMSE for the GKF, IMM, and TCN are all quite close during CV mode, suggesting that both schemes achieve near-optimal RMSE performance. During CT mode, the GKF lower bound is likely rather loose, as knowledge of the true turn rate provides it a considerable advantage during this mode; however, the prediction RMSEs for IMM and TCN are again rather comparable during this mode, with IMM prediction RMSE being just slightly lower than the TCN. As for the case with switching, overall the TCN has lower prediction RMSE than the IMM. This is largely due to the superior RMSE performance of the TCN during CV-to-CT transitions, as we will now show.

Figures 6 and 7 depict the prediction RMSE values before and during the transitions from time steps 0 through 25. While the TCN exhibits a slightly higher RMSE than the IMM during the transition from CT mode to CV mode in Fig. 6, the TCN has significantly lower RMSE when transitioning from CV mode to CT mode as shown in Fig. 7. For the chosen transition probabilities and with the definition of a mode "transition" as lasting 25 samples, the target spent 38.5% of its time in the CV mode, 38.5% in CT mode, and 11.5% in each of the two types of mode transitions.

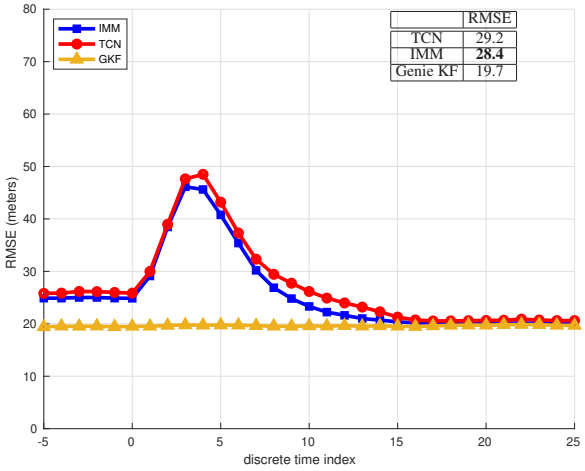


Figure 6. Root Mean-Square Prediction Error During CT-to-CV transitions. RMSE values in table computed over discrete time values $0 \leq k \leq 25$.

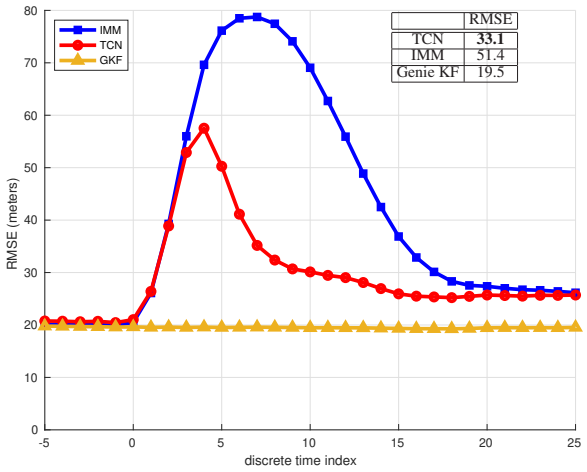


Figure 7. Root Mean-Square Prediction Error During CV-to-CT transitions. RMSE values in table computed over discrete time values $0 \leq k \leq 25$.

In summary, while the TCN is slightly inferior to the IMM in terms of prediction RMSE during long stretches of CV and/or CT modes as well as during transitions from CT-to-CV mode, the ability of the TCN-based approach to more quickly recognize that the target has entered CT mode leads, overall, to a performance advantage over the IMM.

6. Conclusion

In this paper, we applied TCNs to predict states of dynamical systems with model switching. We considered Gilbert-Elliott channels and maneuvering target tracking as examples and demonstrated that the TCN achieves mean-squared prediction error at least as good as classical algorithms while not requiring any knowledge of the system model. Possible future directions include investigating the use of TCNs in other dynamical systems with model switching, such as automotive traffic modeling, power plant control, wireless energy transfer, and scheduling information transfer in communications channels. Because research on TCNs more broadly is advancing at a fast pace, further research could also include recent enhancements to the TCN architecture and training approach, such as rapid training with small amounts of data. Additionally it may be beneficial to compare performance of the TCN to other deep learning architectures to give a more complete performance evaluation. All source code from this paper is available at [25].

Acknowledgments

This work was supported by National Science Foundation awards CNS-1836690, CNS-1836695, and a Research Experiences for Undergraduates (REU) Supplement.

References

[1] O. L. V. Costa, M. D. Fragoso, and R. P. Marques, *Discrete-time Markov jump linear systems*. Springer Science & Business Media, 2006.

[2] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Wiley, New York, 2001.

[3] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[4] E. N. Gilbert, "Capacity of a burst-noise channel," *The Bell System Technical Journal*, vol. 39, no. 5, pp. 1253–1265, 1960.

[5] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *The Bell System Technical Journal*, vol. 42, no. 5, pp. 1977–1997, 1963.

[6] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. part I. dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, 2003.

[7] N. N. Krasovskii and E. A. Lidskii, "Analytical design of controllers in systems with random attributes," *Automat. Remote Contr.*, vol. 22, no. 9, pp. 1021–1025, 1961.

[8] P. Zhao, Y. Kang, and Y. Zhao, "A brief tutorial and survey on markovian jump systems: Stability and control," *IEEE Syst., Man, Cybern. Mag.*, vol. 5, no. 2, pp. 37–C3, 2019.

[9] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems [j]," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[10] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. USA: Prentice-Hall, Inc., 1993.

[11] B. D. Anderson and J. B. Moore, *Optimal Filtering*. Dover Publications, 1979.

[12] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*, 1st ed. Hoboken: Wiley, 2006.

[13] P. A. Ruymgaart, T. T. Soong, and T. S. T., *Mathematics of Kalman-Bucy Filtering*, 1st ed. Berlin Heidelberg New York Tokyo: Springer-Verlag, 1985, vol. 136.

[14] Q. Ge, T. Shao, Z. Duan, and C. Wen, "Performance analysis of the Kalman filter with mismatched noise covariances," *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 4014–4019, 2016.

[15] H. A. P. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with markovian switching coefficients," *IEEE Trans. Autom. Control*, vol. 33, no. 8, pp. 780–783, 1988.

[16] J. K. Tugnait, "Detection and estimation for abruptly changing systems," *Automatica*, vol. 18, no. 5, pp. 607–615, 1982.

[17] Y. Zhang, "Hourly traffic forecasts using interacting multiple model (IMM) predictor," *IEEE Signal Process. Lett.*, vol. 18, no. 10, pp. 607–610, 2011.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[19] J. Yan, L. Mu, L. Wang, R. Ranjan, and A. Y. Zomaya, "Temporal convolutional networks for the advance prediction of ENSO," *Scientific Reports*, vol. 10, no. 1, pp. 1–15, 2020.

[20] W. Zhao, Y. Gao, T. Ji, X. Wan, F. Ye, and G. Bai, "Deep temporal convolutional networks for short-term traffic flow forecasting," *IEEE Access*, vol. 7, pp. 114 496–114 507, 2019.

[21] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[22] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 156–165.

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[24] P. Remy. Keras TCN source code. [Online]. Available: https://github.com/philipperemy/keras-tcn

[25] A. Grootveld *et al.* Source code used in 'Tracking of dynamical processes with model switching using temporal convolutional networks'. [Online]. Available: https://github.com/aspectlab/ModelSwitching

[26] N. Taşpinar and M. N. Seyman, "Back propagation neural network approach for channel estimation in ofdm system," in *IEEE Intl. Conf. on Wireless Comms., Networking and Info. Security*, 2010, pp. 265–268.

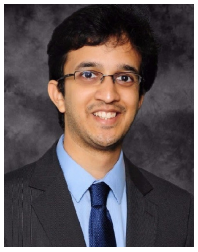
BIOGRAPHY

Arick Grootveld *is currently pursuing a B.S. in Electrical and Computer Engineering at Western Washington University and expects to graduate in June 2021. His current research activities include Radar Signal Processing and Machine Learning for Signal Processing. He is currently applying to graduate schools in hopes of continuing to conduct research in these areas while learning more about signal processing.*

Vlad I. Bugayev *received the B.S. in Electrical Engineering from Western Washington University in June 2020 with minors in Mathematics and Computer Systems. He now resides in Seattle, WA and is looking for employment while simultaneously pursuing the development of audio-visual technologies and learning new technical skills.*

Leah Lackey *is currently pursuing a B.S. in Electrical and Computer Engineering at Western Washington University and expects to graduate in June 2021. Her current research activities include working with convolutional triplet neural networks for assessing paper texture similarity.*

Andrew G. Klein *received the B.S. degree from Cornell University, Ithaca, NY, USA, the M.S. degree from the University of California, Berkeley, CA, USA, and the Ph.D. degree from Cornell University, all in electrical engineering. Previously, he was an Assistant Professor with the Worcester Polytechnic Institute, Worcester, MA, USA, from 2007 to 2014, and he was a Post-Doctoral Researcher with Supélec/LSS, Paris, France, from 2006 to 2007. He joined the Department of Engineering and Design, Western Washington University, Bellingham, WA, USA, in 2014, where he is currently a Professor.*



Kirty Vedula *received the M.S. degree from Rutgers University in 2014, and the Ph.D. degree from Worcester Polytechnic Institute in 2020, both in Electrical and Computer Engineering. He is currently working in the Wireless R&D Division at Qualcomm, San Diego, CA, USA. Previously, he worked as an R&D intern at Philips, Witricity and Mathworks. His research interests are in signal processing, wireless communication systems and machine learning.*

D. Richard Brown III *received the B.S. and M.S. degrees from the University of Connecticut in 1992 and 1996, respectively, and the Ph.D. degree from Cornell University in 2000, all in electrical engineering. From 1992 to 1997, he was with General Electric Electrical Distribution and Control. He was a Faculty Member with the Worcester Polytechnic Institute, Worcester, MA, USA, in 2000. He was a Visiting Associate Professor with Princeton University from 2007 to 2008. From 2016 through 2018, he was with the Computing and Communication Foundations Division, National Science Foundation, as a Program Director.*